FLEX AND PINCH: A CASE STUDY OF WHOLE HAND INPUT DESIGN FOR VIRTUAL ENVIRONMENT INTERACTION

JOSEPH J. LAVIOLA JR. and ROBERT C. ZELEZNIK

Brown University Site of the NSF Science and Technology Center
for Computer Graphics and Scientific Visualization
PO Box 1910, Providence, RI 02912 USA

ABSTRACT

We present a discussion of design issues involving whole hand input in virtual environments. In many cases, whole hand input devices limit the types of interaction that the user can perform in the virtual world due to the nature of the device. One possible approach to alleviate these limitations is to provide hybrid input devices which enable the user to combine information generated from two different whole hand input devices. In this paper, we describe our Pinch Glove like input device which is used as a tool to augment bend-sensing gloves for object manipulation and menu selection as well as a method to test and evaluate different hand postures and gestures that could not be developed with a single whole hand device.

KEYWORDS: Human-Computer Interaction, Virtual Environments, 3D Graphics Applications, Conductive Cloth, Flex and Pinch Input

INTRODUCTION

There have been a number of different approaches for interacting in virtual environments. In general, these approaches have attempted to solve small interface problems in isolation without incorporating them into complete interface solutions. For example, consider the Head Crusher object selection technique[1] which allows the user to very naturally select and manipulate 3D objects with just one hand by positioning the thumb and forefinger around a 2D image of the desired object. To actually use this technique for object selection, the user must hold and press a button in their other hand.

Another important reason why many of these interaction techniques solve small problems in isolation has to do with the nature of the available input devices used. In most cases, individually specialized input devices work well for the interaction techniques they were designed for. However, they have difficulty mapping combinations of techniques or applying different techniques due to their inflexibility. For example, consider bend-sensing gloves which report continuous joint angle measurements of the fingers. With these devices, relatively slow and complicated posture and gesture recognition techniques must be used to generate discrete events that would otherwise be trivial with a button press.

In order to increase the flexibility of input devices, to extend existing virtual environment interaction techniques, and to create more robust virtual environment interfaces, we believe that hybrid interfaces – interfaces that seamlessly combine input devices and interaction techniques – will provide a more flexible and robust method of interacting in virtual environments. With Flex and Pinch input, we have developed a hybrid input device which combines continuous joint angle measurements and discrete pinch button input. By having this combination, we can improve on a number of existing virtual environment interface techniques and develop new ones.

ORGANIZATION

The remainder of this paper is organized in the following manner. The next section describes previous work related to Flex and Pinch followed by a description of the components and design issues in developing our interface hardware. Then we describe a number of interface techniques that use Flex and Pinch input. Finally, the last two sections provide areas for future work and a conclusion.

PREVIOUS WORK

There are two basic approaches to using whole hand input in virtual environments. First, the non-invasive approach uses vision-based tracking[2] so the user is not physically attached to the computer. Second, the invasive approach uses a glove-based device or devices to extract information from the hands. In each approach, we can extract two different types of data, namely geometrical data and topological data. Geometrical data represents information about the hand's shape while topological data provides information about how the fingers touch each other and other parts of the hand. Although a non-invasive approach maybe preferred, it is difficult to extract both geometrical and topological information due to problems with computer vision such as occlusion. Therefore, we focus on the invasive approach.

With the invasive approach, two types of glove-based input devices have been developed. The first, bend-sensing gloves[3][4][5] , measure finger joint movement, and second, the Pinch Glove[6][7], detect electrical contacts between each of the finger tips. Unfortunately, bend-sensing and pinch gloves have faults when used in isolation. Bend-sensing gloves are good at extracting geometrical information which enables them to represent the user's hands in the virtual environment. They can be used to mimic interface widgets such as sliders and dials[8], but do not have useful methods for signaling the activation or deactivation of the widget. Bend-sensing gloves are also used in conjunction with hand posture and gesture recognition, but it can be difficult to determine when one gesture begins and another ends without applying constraints to the users gesture space[9]. Conversely, Pinch gloves provide a series of button widgets that are placed on each finger tip which allows for the extraction of topological data for interactions such as pinching postures. However, they have no way of determining the flexing of the fingers and they make it difficult to represent the hand in a virtual environment.

There have been few attempts to combine the two types of information that each type of data glove provides. With the exception of Grimes' Digital Data Entry Glove which was developed specifically for entering text using the Single Hand Manual Alphabet[10], there has been little work done with combining discrete and continuous whole hand input devices to extract both geometrical and topological data simultaneously.

OVERVIEW OF APPROACH

In order to develop an interface that combines both geometrical and topological data, we built a hardware prototyping system for testing and evaluating different interface designs[1]. The hardware system provides a number of benefits in that it employs a plug and play strategy for quickly adding and removing button widgets or their components. Our system enables users to incorporate up to 16 cloth sensors in a wearable interface. Conductive cloth[11] sensors provide two important functions: first, each sensor knows when it comes in contact with another sensor and specifically which other sensor it contacts, second, the nature of the cloth lends itself for use on gloves or clothing.

Using our prototyping system, we have constructed a device based on the Fakespace Pinch Glove[6]. As a hardware input device, it provides more functionality than the Pinch Glove since it uses eight cloth buttons instead of five which allows for more button combinations. In general, five of these cloth buttons can be placed around each of the finger tips, while the other three can be placed arbitrarily about the hand[2]. Using this device, we augment existing bend-sensing gloves to create Flex and Pinch input (see Figure 1).

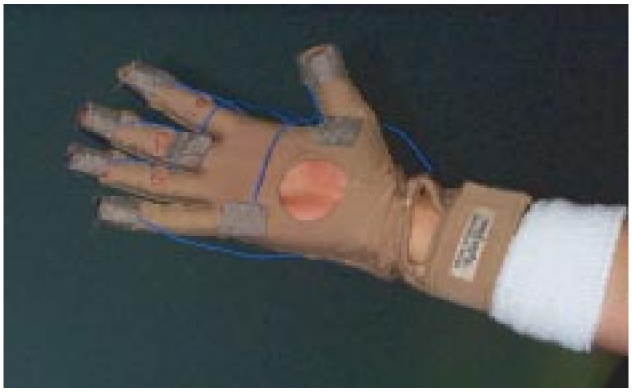

Figure 1: The Flex and Pinch input system. Although a CyberGlove[4] is shown, any bend-sensing glove can be used.

INTERACTION TECHNIQUES USING FLEX AND PINCH INPUT

With Flex and Pinch input, we can improve on a number of existing techniques for selecting objects in virtual environments and create new techniques that could not be developed without the combination of geometrical and topological data. For example, one of the major problems with the image plane interaction techniques such as the head crusher, sticky finger, lifting palm, and framing hands object selection techniques[1] is that the user cannot activate the selection with the primary hand. As a result, the user requires an additional, separate input device for triggering the selection operation.

Flex and Pinch input provides a simple yet effective and seamless method for starting and stopping object selection by placing the cloth buttons in appropriate places on the users primary hand. For example, with the head crusher technique, we can place the cloth buttons on the thumb and middle finger so when the user positions the thumb and forefinger around the object a middle finger to thumb contact will signal the object should be selected. Another button press would signal the release of the object. The cloth contacts can be placed in other positions such as on the middle finger and on the palm by the base of the thumb or on the right side of the index finger and the left side of the middle finger. In a similar manner, cloth contacts are placed on

[1] Appendix A provided a description of the electronics and the various components used for building our hardware system.

[2] This presents one of many possible combinations for placement of the cloth buttons. The device could have be worn with anywhere from two to 16 cloth buttons of any shape or size. This presents a clear advantage over other inflexible input devices.

the hand for the sticky finger and lifting palm techniques to start and stop object selection while cloth contacts are placed on both hands for the framing hands selection technique. Figure 2 shows the Head Crusher technique with placement of the cloth contacts between the forefinger and middle finger.

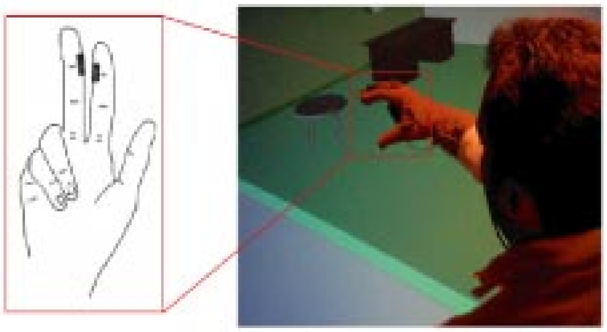

Figure 2: A user wearing the Flex and Pinch input device is about to invoke the Head Crusher object selection technique on a round table. By placing his middle and index finger together, the user can activate the selection operation and move the table.

Another method that has been used for selecting objects in virtual environments is to cast a laser into the scene from the users hand to select a given object[12]. As with the image plane techniques, the problem with laser pointing is it is difficult to start and stop the selection with only one input device. For example, one laser pointing object selection method uses a point and clutch posturing mechanism to select objects in a virtual environment where clutching is performed by flexing the thumb[13]. The problem with using this clutching mechanism is that in order to achieve robust recognition, the user must make postures using extreme configurations of the hand which puts undo strain on the two tendons in the thumb. Using Flex and Pinch input we can alleviate this problem by placing cloth contacts on the thumb and on the right side of the middle finger as shown in Figure 3. This provides a much more natural movement and puts no strain on the thumb tendons[3].

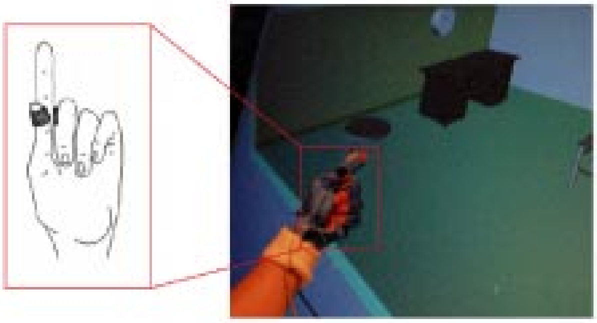

Figure 3: A user pointing at and selecting a desk in the virtual environment. The user makes the selection by pressing the thumb to the right side of the middle finger.

Bend-sensing gloves have the capability of being used as analog sliders since these gloves report continuous measurements of the joint angles in the hand. However, used in isolation, it can be difficult to determine when the user wants to actually use one of the fingers as a slider widget. Using Flex and Pinch input, a seamless transition between the discrete events from the cloth contacts and the continuous updating from the bend sensors can be made which provides a mechanism for activating and deactivating the sliders when needed. For example, we can cycle through menu items with a finger[4]. A button press creates the menu and as the button is held, the user can cycle through the menu items by flexing or extending a finger. If the user does not wish to select an item, they need to release the button when their finger is fully extended or fully flexed. We are currently exploring how many menu items a user can easily invoke using this technique. Using the same configuration we also can change an object's scaling, translational, and rotational parameters.

Finally, an important benefit of using the pinch component of Flex and Pinch is that it gives application developers a method to test out different hand postures and gestures. In many cases, when a developer wants to test a new hand posture or gesture, they have to retrain their gesture recognition algorithms[14] which is time consuming. The pinch component of Flex and Pinch input allows the developer to quickly move cloth contacts from one part of the hand to another without having to change any software components or restart the application. This allows the application developer to quickly test the feeling and ergonomics of certain hand postures and gestures. Also, with the ability to move the cloth contacts anywhere on the hand, we can create whole hand interfaces that could not be implemented with a bend-sensing glove or the Pinch Glove used in isolation.

FUTURE WORK

There are a number of areas of future work that must be researched to determine if these hybrid interfaces provide virtual environment interaction methods that are useful. We plan to continue developing new hybrid input devices and exploring how they can give users better performance in virtual environment applications. In order to do this, extensive user studies are required to evaluate whether our

[3] One could argue that the user could make a posture that is identical to the user's hand configuration when using Flex and Pinch input. However, hand gesture and posture recognition is not perfect, and if the hardware is working properly, the pinching mechanism will provide 100 percent accuracy.

[4] In this case, one cloth contact is placed on the thumb while the second is placed on the left side of the forefinger between the Proximal Interphalangeal and Metacarpophalangeal joints.

interaction techniques are indeed better than existing techniques. We believe that the input devices and interaction techniques we have developed are just the tip of the iceberg. As a result, it is important to continue to research how they can be applied to different interactions in and out of virtual environments.

Another area of work that needs further exploration is whether or not users prefer a wearable interface solution over a less obtrusive solution such as computer vision-based interaction. Although the invasive approach provides more functionality since occlusion problems can occur with vision-based tracking, typically users do not want to be physically connected to the computer. One compromise between unobtrusive interfaces and increased functionality is using wireless input devices.

CONCLUSION

In this paper we have presented a case study on whole hand input design issues for virtual environment interaction using Flex and Pinch input. Using our custom built hardware prototyping system, we have developed a multipurpose button based input device that can be used to develop seamless, hybrid interfaces by augmenting devices that produce continuous input events. With Flex and Pinch input, we can improve on existing virtual environment interaction techniques such as the image plane object selection techniques[1]. We also can develop novel hand postures and gestures that could not otherwise be developed with a device that generates purely geometrical or topological data. With further study and research, it is our goal to make the geometrical/topological approach a powerful metaphor for interaction in virtual environments.

ACKNOWLEDGMENTS

Special thanks to Timothy Rowley for helpful discussions during the hardware design and implementation, Brian Perkins for providing the electronics implementation, and Christine Waggoner for invaluable assistance with creating Flex and Pinch. This work is supported in part by the NSF Graphics and Visualization Center, International Business Machines, Advanced Networks and Services, Alias/Wavefront, Autodesk, Microsoft, Sun Microsystems, and TACO.

APPENDIX A

This appendix provides information on the design and implementation of our custom built hardware for quickly prototyping and testing hybrid, whole hand input devices.

DESIGN AND IMPLEMENTATION OF ELECTRONICS

The Microchip PIC processor[15] was chosen as the primary means of interfacing the touch sensors with the rest of the system. The low cost and simple programming of these chips made them suitable for the task. The 16C63[16] provided a UART for serial communications with the workstation, and enough I/O pins to allow the touch sensors to be monitored without extra glue logic. The output pins of the micro-controller were protected from electrostatic discharge with a resistor capacitor network. Additionally, an rs232 driver chip was needed to step the five volt output of the PIC to rs232 line levels.

All 163 possible non-redundant contact possibilities between pairs of wires are reported by separate keycodes. It is up to the microcode driver to report separate keycodes for wire connections while the driver on the workstation infers contacts between more than two wires. For example, if contacts one, two, and three are touching, the microcontroller will report that one and two are touching by issuing one keycode, one and three are touching by issuing another keycode, and that two and three are also touching by issuing a third keycode. It is up to the driver software to determine that there are actually three wires that are all touching. This lowers the amount of memory needed on the microcontroller, and makes the software simpler and faster.

PARTS LIST

PART	USAGE
PIC16C63	8 bit microcontroller with built in UART primary interface chip
16x 20K ohm resistors	pull up resistors
16x 2K ohm resistors	protection resistors
16x 1000pF capacitors	protection capacitors
LT1081	RS232 driver/receiver converts 5 volt PIC output to RS232 levels

ELECTRONICS PSEUDOCODE

This peudocode represents the code for the PIC on the electronics box. Each short possibility has a byte allocated to it to represent the status (short or unshort) and a timer to determine whether the short has lasted long enough to transmit. This implementation cuts down on noise and bouncing problems.

Algorithm *1*

```
initializeMemory()
for each pin
    do set a voltage on pin;
        for each (otherpin > pin)
            do check for voltage on otherpin;
                if (pin status changed)
                    increment keycode timer;
                if (timer expired)
                    set keycode status;
                    transmit status change;
```

REFERENCES

[1] J.S. Pierce, A.S. Forsberg, M.J. Conway, S. Hong, R.C. Zeleznik, and M.R. Mine. Image Plane Interaction Techniques in 3D Immersive Environments. *Proceedings of the 1997 Symposium on Interactive 3D Graphics*, 1997, 39-44.

[2] Y. Kuno, T. Ishiyama, K. Jo, N. Shimada and Y. Shirai. Vision-Based Human Interface System: Selectively Recognizing Intentional Hand Gestures. In Proceedings of the *IASTED International Conference on Computer Graphics and Imaging*, 1998, 219-223.

[3] Nissho Electronics Corporation. Introduction to SuperGlove. Tokyo, Japan, 1997.

[4] Virtual Technologies. CyberGlove$^{\text{TM}}$ User's Manual. Palo Alto, California, 1993.

[5] T.G. Zimmerman, J. Lanier, C. Blanchard, S. Bryson, and Y. Harvill. A Hand Gesture Interface Device. *In Proceedings of CHI+GI'87 Human Factors in Computing Systems and Graphics Interface*, 1987 189-192.

[6] Fakespace Pinch$^{\text{TM}}$ Glove System Installation Guide and User Handbook, Mountain View, California, 1997.

[7] D.J. Mapes and M.J. Moshell. A Two-Handed Interface for Object Manipulation in Virtual Environments. *PRESENSE Teleoperators and Virtual Environments*, 1995, 4(4):403-416.

[8] D.J. Sturman and D. Zeltzer. A Survey of Glove-based Input. *IEEE Computer Graphics and Applications*, 1994, 14(1):30-39.

[9] D.J. Sturman. *Whole Hand Input*. Ph.D. dissertation, Massachusetts Institute of Technology, 1992.

[10] G. Grimes. Digital Data Entry Glove Interface Device. Bell Telephone Laboratories, Murray Hill, New Jersey. US Patent Number 4,414,537.

[11] S. Mann. Smart Clothing: The Wearable Computer and WearCam. *Personal Technologies*, Volume 1, Issue 1, March, 1997.

[12] K. Hinkley, R. Pausch, J.C. Goble, and N.F. Kassel. A Survey of Design Issues in Spatial Input. *Proceedings of the ACM Symposium on User Interface Software and Technology*, 1994, 213-222.

[13] J.J. LaViola Jr. A Multimodal Interface Framework For Using Hand Gestures and Speech in Virtual Environment Applications. To appear in *Lecture Notes In Artifical Intelligence: The Gesture Workshop'99*, Springer-Verlag, 1999.

[14] S. Fels and G. Hinton. Glove-TalkII: An Adaptive Gesture-to-Format Interface. *Proceedings of CHI'95 Human Factors in Computing Systems*, 1995, 456-463.

[15] http://www.microchip.com/10/Lit/PICmicro/index.htm.

[16] http://www.microchip.com/10/Lit/PICmicro/16C6X/index.htm.